1. A software development system for applications that run on a data network which couples a server computer and a client computer, wherein the client computer runs a browser program, comprising

a page generator capable of generating functional application pages with additional editing features for interpretation by the browser program;

an editor capable of directly operating on the pages displayed by the browser thereby allowing the user to work on a functional application during development.

2. A software development system as claimed in claim 1, further comprising a plurality of components, and wherein developed applications comprise at least one page template capable of containing components, and wherein the editor provides features to insert, modify and delete components on page templates.

3. A software development system as claimed in claim 2, wherein at least one of the components reacts interactively on user input by executing instructions on the server.

4. A software development system as in claim 3, wherein at least one of the components contains at least one other component.

5. A software development system as in claim 3, wherein the set of components on pages generated from a single page template can vary for different requests of the same page template.

6. A software development system for use in a data network which couples a server computer to a client computer, wherein the client computer includes a first software program for generating a request for one or more pages from the server computer and for displaying pages, and wherein the server computer includes a second software program for receiving and processing the request from the client computer, for generating and storing pages, and for transmitting pages to the client computer in response to requests, the server computer further comprising:

a data store,

a plurality of components residing in the data store, including components that react interactively on user input by executing instructions on the server;

a plurality of page templates residing in the data store, at least one page template having at least one selected component incorporated therein; and

a server processor controlled by a third software program, said program providing instructions for selecting a page template based on the request from the client computer and instructions for generating a page from the page template for transmission to the client computer.

7. The development system of claim 6, further comprising a component editor controlled by a fourth software program, said program providing instructions for interactively editing selected components on a page template.

8. The development system of claim 6, wherein a component is nested within a component.

9. A method for generating documents for display by a browser using components that react interactively on user input by executing instructions on a server, comprising the following steps for execution on the server upon a document request:

assigning a unique identifier to at least one of the components; and

embedding the unique identifier into a generated document.

10. The method of claim 9, further comprising storing data on the server representing at least one of the components.

11. The method of claim 10, further comprising:

analyzing the request sent by the browser for unique identifiers; and

calling a function for the interactive components referenced by at least one of the unique identifiers contained in the request.

12. The method of claim 11, wherein at least one of the components is contained on a document template.

13. The method of claim 11, wherein at least one of the components is called by a program.

14. The method of claim 11, wherein at least one of the components is called by a another component.

15. The method of claim 11, wherein the data is stored into an object of an object oriented programming language and wherein the function is a method of the object.

16. A method for implementing client server applications, comprising:

storing data objects on a server and assigning a unique identifier to each data object;

dynamically generating a document with the unique identifier embedded in the document; and

analyzing requests for unique identifiers and calling at least one function for a data object associated with one of the unique identifiers found in the request.

17. The method of claim 16, wherein the unique identifier is embedded inside a uniform resource locator contained in a tag of the document.

18. The method of claim 16, wherein the unique identifier is embedded in scripts contained in the document.

19. The method of claim 16, wherein the unique identifier is unique within a single session.

20. The method of claim 16, wherein the unique identifier is unique within all documents generated by a single server within a defined time period.

21. The method of claim 16, wherein the data objects are created by an object-oriented programming language and said function is a method of one of these objects.

22. A computer running an application to develop and maintain applications using a web browser, comprising:

an editor operable within the web browser for inserting, deleting, and modifying components on document templates; and

a document generator for processing document templates and for generating documents from the document templates that are understandable by the web browser.

23. A computer as in claim 22, wherein the editor operates functional applications in an edit mode permitting editing directly in the web browser.

24. A computer as in claim 23 wherein at least one of the components can react on subsequent document requests containing user responses by executing selected instructions.

25. A computer as in claim 24, wherein the computer further comprises:

a store of component classes, each component class implementing one component kind; and

a parser able to detect components marked on document templates;

wherein the document generator works upon a document request using component classes to generate browser code; and

wherein the editor is capable of showing a menu of components for insertion into the document templates.

26. A system to modify documents on a server in a data network which couples said server computer to a client computer, the server computer comprising:

a document store;

a first software program including instructions for transforming a first document retrieved from the document store into a second document having features which permit editing of the first

document such that at least a part of the second document appears and functions similar to the first document; and

a second software program including instructions to receive information from the client computer and instructions to modify documents stored in the document store.

27. The system of claim 26, wherein the second document includes at least one component and at least one handle to indicate the position of the component to the user.

28. The system of claim 27, wherein the second document includes handles and choosing one of the handles selects an editing operation

29. The system of claim 28, wherein at least one handle indicates the position of at least one component contained in the first document and said editing operation is chosen from the group of modifying the component, deleting the component, displaying information regarding the component, and inserting a new component.

30. The system of claim 26, wherein the features are scripts.

31. The system of claim 30, wherein the scripts are generated specifically for the second document and encapsulate information which is incorporated into the first document.

32. The system as in claim 26, wherein the features incorporate information regarding the first document into the second document.

33. A system as in claim 32, wherein the information incorporated into the second document is used on the client computer in order to send change requests for the first document to the server.

---

34. A method for generating a document for display in a browser from a document template containing components, comprising:

for each component denoted on the document template, identifying a component class of the component; and

based on data contained in a request initiated by the browser storing a first object of the component class, the first object representing the component.

35. The method of claim 44, further comprising calling a method of said component class to generate browser code, said method being the constructor.

36. The method of claim 34, further comprising, for all components having a name attribute, looking up the component object in session memory based on said name attribute.

37. The method of claim 34, further comprising, for at least one component kind, for all components denoted on the document template having said component kind;

generating a unique identifier;

assigning said unique identifier to said object, and

embedding said unique identifier into the browser code.

38. The method of claim 37, further comprising:

inserting objects for the components of at least said component kind into a list of listening components;

working through all objects stored in the list of listening components whose unique identifier occurs inside a name in the form data set; and

calling a method of at least one of these objects.

39. The method of claim 34, wherein the document template is parsed into a list of nodes, including text and component nodes, said method further comprising:

determining if the current node is text or a component;

if component, then calling a method for the component, comprising:

evaluating the attributes of the component if necessary;

identifying the component class associated with the component; and

calling the constructor method of the component class,

said constructor method generating browser code;

if text, then generating the text; and

repeating these steps for each node.

40. The method of claim 39, wherein at least one component contains nested components and the method of claim 39 is recursively performed for all nodes nested inside the component.

41. A software development system as in claim 1, the editor comprising a client part for execution on the client computer.

42. A software development system as in claim 41, wherein the client part comprises instructions that are automatically downloaded from the server prior to editing.

43. A system as in claim 26, additionally comprising at least one script for automatic download to the client that works in cooperation with the second document to permit editing of the first document.

44. The method of claim 34 wherein storing the first object comprises creating a new object as necessary.

45. The method as in claim 34 wherein components are denoted on document templates using tag syntax.

46. The method as in claim 45 wherein the tag name identifies a component class.

47. The method as in claim 36 wherein components are denoted on document templates using tag syntax, wherein the tag name identifies a component class.

48. The method as in claim 36 wherein the component object, if found, is reused to store the first object.

49. The method as in claim 36 wherein in case a component has a name attribute but no component object is found a new object is created and stored under said name in session memory.

50. The method as in claim 49 wherein new objects are created for all components not having a name attribute.

51. A system for editing components on web document templates for use with a first software program including first instructions for generating a document request to obtain at least one generated document from a second software program and for displaying the generated document, the second software program capable of receiving and processing a document request and of transmitting first documents to the first software program in response to requests, said system comprising:

a plurality of components,

a plurality of document templates,

the second software program transmitting, while processing selected requests, second documents to the first software program that make the first software program display a user interface for editing functions used for maintaining components on document templates,

a third software program used by the second software program while processing selected document requests, the third software program including third instructions for modifying document templates in order to perform said editing functions.

52. The system of claim 51, wherein components include fourth program instructions including steps to generate browser code prior to transmission to the first software program.

53. A system in claim 52 running on a data network, coupling a server computer and a client computer, the first program running on the client computer, the second program running on the server computer.

54. A system in claim 52 wherein second documents are HTML pages with embedded scripts.

55. The system of claim 52, wherein the edit function includes adding a component to a document template, removing a component from a document template, and modifying attributes of a component on a document template.

56. The system of claim 52, further comprising a fifth software program used by the second software program while processing selected document requests, the fifth software program including fifth instructions for generating generated documents from document templates thereby calling fourth program instructions.

57. The system of claim 56, wherein the generated document includes, if requested in edit mode, edit features for interpretation by the first software program.

58. The system of claim 56 further comprising instructions to allow the user to click on the generated document to select items to perform edit functions on.

59. A software development system for dynamic web documents comprising:

an editor program for editing dynamic web documents,

a document generator for generating generated documents from dynamic web documents,

the editor program comprising first instructions for requesting the document generator to process a dynamic web document leading to a generated document,

the editor program further comprising second instructions for displaying at least some information items contained on said generated document in a view which allows the user to select an item to which a modification function will be applied,

the editor program further comprising third instructions to modify the dynamic web document to perform said modification function.

60. The software development system of Claim 59 running on a data network, which couples a server computer and a client computer, the document generator running on the server computer the editor at least partly running on the client computer.

61. The software development system of claim 60, wherein the document generator further comprises fourth instructions for execution during document generation to collect edit-information for use by the editor.

62. The software development system of claim 60, wherein the editor uses a web browser for displaying said view.

63. The software development system of claim 60, able to automatically repeat requesting the document generator to process the dynamic web document if required.

64. The software development system of Claim 59 further comprising a plurality of components marked on the dynamic web document, components including instructions for use by the document generator to generate browser code.

65. The software development system of claim 64, wherein the editor uses a web browser for displaying said view.

66. The software development system of claim 64, wherein modification functions include insert of a component, delete of a component, and modify attributes of a component.

67. The software development system of claim 59, wherein said view looks, except for editing features, similar to the end-user view of the generated document.

68. The software development system of claim 59, wherein the document generator further comprises sixth instructions to collect edit-information for use by the editor, said sixth instructions for execution during document generation.

69. The software development system of claim 68, wherein the editor uses the edit-information to correctly modify the dynamic web document.

70. The software development system of claim 69, further comprising a plurality of components marked on the dynamic web document, wherein the edit-information comprises position information on the components contained in the document template.

71. The software development system of claim 59, wherein the editor uses a web browser for displaying said view.

72. The software development system of claim 71, wherein first instructions comprise seventh instructions for initiating a reload in the browser.

73. The software development system of claim 59 the editor program further comprising eighth instructions to display information on at least one element of the dynamic web document, that is replaced during document generation, without requesting the document generator to regenerate the generated document.

74. A software development system for document templates that are intended for transformation into generated documents for display by a first software program, the first software program including first instructions for generating a document request to obtain at least one generated document and for displaying the generated document, comprising:

a plurality of components, that include instructions to generate browser code prior to transmission to the first software program,

an editor capable of performing edit functions maintaining components on document templates, the components capable to cooperate with the editor,

a plurality of document templates having components denoted thereon, and

a document generator comprising second instructions to, upon a document request, generate generated documents from document templates for display by the first software program wherein the set of components on the generated document can vary for different document requests for the same document template.

75. The software development system as in claim 74, wherein edit function comprises adding a component, modifying a component, and deleting a component.

76. The software development system as in claim 74, wherein tag syntax is used to denote components on document templates, whereby the tag name identifies the component kind.

77. The software development system of claim 74 running on a data network, which couples a server computer and a client computer, the document generator running on the server computer the editor running, at least partly, on the client computer.

78. The software development system as in claim 74, wherein a component, that can react interactively on subsequent document requests, can be excluded from the generated document.

79. The software development system as in claim 78 further comprising third instructions to prevent excluded components from reacting on subsequent document requests.

80. A software development system as in claim 79, third instructions comprising fourth instructions to, upon a first document request, store information in session memory on all components, that are present on the generated document, and fifth instructions to, upon subsequent document requests, only react on components that have been remembered in session memory thereby avoiding tampering with excluded components on the side of the first program.

81. A software development system as in claim 74 wherein at least one document template has a first and a second component denoted thereon in a way that the first component contains the second component, the first component containing sixth instructions to decide about exclusion of the second component from the generated document.

82. A software development system as in claim 74 the editor able to provide an editable view taking the varying set of components into account.

83. A software development system as in claim 74 the editor able to provide an editable view that includes and excludes selected components similarly as the final application.

84. A software development system as in claim 74 wherein the generated document contains more components than the document template for at least one document request.

85. The software development system as in claim 74, wherein multiple instances of a third component denoted on the document template can be included in the generated document.

86. The software development system as in claim 85, further comprising seventh instructions to assign a unique identifier to each component instance, whereby the third component includes eighth instructions to qualify names generated into the browser code with the unique identifier.

87. A software development system as in claim 74, wherein at least one document template has a fourth and a fifth component denoted thereon in a way that the fourth component contains the fifth component, the fourth component containing ninth instructions to decide about how many instances of the fifth component are included into the generated document.

88. A software development system as in claim 74 the editor able to provide an editable view that include multiple instances of components similarly as the final application.

89. A software development system as in claim 74 wherein at least one sixth component includes tenth instructions to display the sixth component, the tenth instructions being used to generate browser code for displaying the sixth component during editing as well as during normal use of the component.

90. An editor for use with a web browser, the editor allowing the user to edit a document displayed by the browser, wherein clicking on said document displayed in the browser initiates editing functions, and scripts contained in said document remain functional, the editor including a first software program for execution within the browser and for processing the clicking on said document.

91. The editor as in claim 90 using at least two windows, a first browser window displaying said document and a second window for displaying information on an element contained in said document.

92. The editor in claim 90 further comprising a second software program for modifying documents in cooperation with the first software program.

93. The editor as in claim 92 further comprising a third program for transforming the document into a generated document thereby adding editing features, the browser displaying the generated document looking similar to the original and interpreting the editing features.

94. The editor as in claim 93 wherein said document is a dynamic document having components denoted thereon, the third software program further comprising instructions for generating browser code for components.

95. The editor as in claim 94 wherein the browser together with the first software program is running on a client computer connected to a server computer via a data network, wherein the second and the third software program run on the server computer.

96. The editor in claim 90 wherein links contained in said document stay functional allowing the user to browse and edit at the same time.

97. A system for displaying dynamically generated documents in a data network coupling a server computer to a client computer, wherein the client computer has a first software program including first instructions for generating a document request to obtain at least one generated document from the server computer and for displaying the generated document, comprising:

a plurality of components for execution on the server computer, including a first component including second program instructions to generate browser code and third program instructions for execution on the server which are initiated by the user interacting with the first component, and,

fourth program instructions on the server computer for, based on data contained in a document request initiated by the first software program on the client computer, generating generated documents for transfer to the client computer and display by the first software program, thereby calling second program instructions of components.

98. The system of claim 97, the server computer further comprising fifth program instructions for analyzing said data and for calling third program instructions of first component as necessary.

99. The system of claim 98, further comprising a plurality of document templates residing in the data store, at least one of the document templates having at least one second component denoted thereon.

100. The system of claim 99, wherein tag syntax is used to denote the second component, whereby the tag name identifies a component.

101. The system of claim 99, wherein at least one third component denoted on a document template contains the first component.

102. The system of claim 101, wherein third components implementation scheme includes logic to decide how often to insert the first component into the generated document.

103. The system of claim 98, the server computer further comprising sixth program instructions for, based on the document request, deciding to insert more than one instance of the first component into the generated document.

104. The system of claim 103, making sure that multiple instances of the first component do not interfere by qualifying names generated into the browser code using unique identifiers.

105. The system of claim 98, the server computer further comprising seventh program instructions for, based on the data, deciding to exclude the first component from the generated document.

106. The system of claim 105, wherein fifth instructions call third instructions only if the first component was contained on a page previously transferred to the client.

107. The system of claim 98, wherein fifth program instructions include eighth program instructions to analyze said data for user interactions with multiple components and to call third program instructions of multiple components as necessary.

108. The system of claim 107, wherein components include ninth instructions to check for errors and fifth instructions include tenth instructions to call ninth instructions of components as necessary and to suppress subsequent calling of third instructions in case of errors.

109. The system of claim 98, further comprising eleventh program instructions for storing a data object in session memory representing at least one component instance included in a generated document, said eleventh program instructions for execution while dynamically generating a document.

110. The system of claim 109, wherein third program instructions are encapsulated in a method of data objects, fifth program instructions including twelfth program instructions for identifying the data object that represents a component instance the user interacted with and for calling said method of said data object as necessary.

111. The system of claim 110, further including thirteenth instructions for deciding based on said data to include more than one instance of a component into the generated document.

112. The system of claim 109, further comprising twelfth program instructions for assigning a unique identifier to the first component instance, for associating the unique identifier with the data object, and for including the unique identifier into the generated document, said twelfth program instructions for execution while dynamically generating a document .

113. The system of claim 112, wherein fifth program instructions analyze said data for unique identifiers and include thirteenth instructions for identifying the associated data object.

114. A system for displaying dynamically generated documents in a data network coupling a server computer to a client computer, wherein the client computer has a first software program including first program instructions for generating a request to obtain at least one generated document from the server computer and for displaying the generated document, comprising:

a plurality of components for execution on the server, at least one of the components including first features to cooperate with an editor in editing said component and second program instructions to generate browser code, and

third program instructions on the server for, based on the data contained in a request initiated by the client computer, generating generated documents for transfer to the client computer, thereby calling second program instructions of components.

115. The system of claim 114 wherein first features include fourth program instructions for passing information to the editor.

116. The system of claim 115 wherein at least part of said information is collected during execution of the components on the server.

117. The system of claim 115 wherein said information is transmitted from the server to the client.

118. The system of claim 115 wherein said information includes attributes of said component.

119. The system of claim 114 wherein first features include fifth instructions that display additional editing features of the component during editing.

120. (new) The system of claim 119 wherein said editing features include handles.

121. The system of claim 114 wherein first features include an extension for use by the editor, said extension for enabling editing of the components attributes.

122. The system of claim 121 wherein said extension is a page for editing components attributes.

123. The system of claim 114 wherein components are denoted on document templates using tag syntax, whereby the tag name identifies a component.

124. The system of claim 114 containing at least one component wherein second program instructions are used to generate browser code for displaying the component during editing and during normal use.

125. A method for editing an application that is built using components and that operates by generating documents comprising the steps of:

running the application, thereby generating a generated document,

displaying a view of the generated document,

selecting a component by clicking on selected portions of said view,

identifying the selected component in the source code of the application,

initiating a modification function modifying the source code of the application.

126. The method of claim 125 wherein the running step and the displaying step are repeated after applying a modification function.

127. The method of claim 125 further comprising collecting edit information for use by the identifying step.